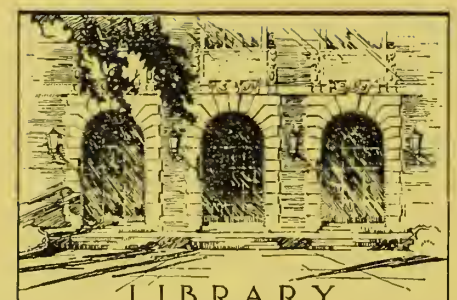

Report No. 264

LOGICAL DESIGN OF AN OPTIMAL NETWORK BY INTEGER LINEAR PROGRAMMING - PART I

by

Saburo Muroga
Toshihide Ibaraki

July 18, 1968

DEPARTMENT OF COMPUTER SCIENCE · UNIVERSITY OF ILLINOIS · URBANA, ILLINOIS

Report No. 264

LOGICAL DESIGN OF AN OPTIMAL NETWORK BY INTEGER LINEAR PROGRAMMING - PART I

by

Saburo Muroga
Toshihide Ibaraki

July 18, 1968

SUMMARY

Logical design of an optimal network by integer linear programming is formulated. Various restrictions on a network are easily incorporated and how to incorporate them is discussed. The number of gates, the number of levels or one of other parameters of a network can be minimized. A network of multiple outputs is also easily treated. The formulation is based on threshold gates, since threshold gates are the generalization of conventional switiching gates, such as AND, OR and NAND gates. The formulation of designing a network of conventional gates can be easily derived from this general formulation. As an example, design of an optimal network with NOR gates is discussed in some detail. Computer programs to design optimal networks with NOR gates for all Boolean functions of three variables are run. Computation time is much less than the exhaustive method of finding optimal networks. The design by integer linear programming is found computationally feasible.

1. Introduction

Logical design of an "optimal" digital network has been one of the most important objectives of switching theory. Optimality of a network is the minimization of the number of gates, connections, levels or whatever the designer wants to minimize, under network restrictions such as fan-ins and fan-outs restrictions. But the design of an optimal network has remained an unsolved problem since it is difficult to formulate the concept of optimization within the framework of Boolean algebra which has been a major mathematical tool in switching theory.

With the advent of integrated circuits and large scale integration the traditional optimality of a network has been further compounded by many engineering considerations and restrictions. Thus within the field of design automation there is a great need for a flexible optimization technique which will allow the designer to specify a variety of composite optimization criteria. Such a technique would not only reduce the cost of digital networks but also reduce the complexity and increase the reliability.

In this paper, the problem is approached by means of integer linear programming. A preliminary computational result is also outlined.

The integer linear programming approach is flexible to incorporate

a wide variety of constraints* which are to be imposed on a network to be designed and to use any of various types of criteria. Another important feature of the approach is that incompletely specified functions can be treated with no essential difference from the case of completely specified functions, contrary to the Boolean algebraic manipulation. A multiple output network can also be handled without any major modification.

An integer linear programming problem in general is stated as follows:

$$\begin{aligned} &\text{minimize} && \vec{c}\,\vec{y} \\ &\text{subject to} && A\vec{y} \geq \vec{b} \\ & && \vec{y} \geq o \ , \end{aligned} \tag{1.1}$$

where a subset of the variables $\vec{y}$ are constrained to be integers. $\vec{c}$ is an N-dimensional vector, $\vec{b}$ is an M-dimensional vector and $\vec{y}$ is an N-dimensional vector of variables. A is an M x N coefficient matrix. N is the number of variables and M is the number of inequalities.

---

* In contrast to our approach, design procedures which have been known to date have no flexibility in using different constraints. For example, the well known design approach by the minimization of a prime implicant expression leads to an optimum network only when the network is to have two levels. (Integer linear programming approach which is completely different from ours in the present paper can be partly used[6]). Gimpel's approach[9] leads to an optimal network only when the network is to have three levels.

The above integer linear programming problem is called all-integer integer linear programming, or mixed-integer integer linear programming, depending on whether all the variables are constrained to be integers or not. Each variable may be further specified to assume general non-negative integral values or binary integral values which are 1 or 0.

There exist basically a few algorithms for integer linear programming. The cutting plane method by Gomory[11] [12] may be applied to general problems of both all-integer and mixed-integer integer programmings. Another is the implicit enumeration method or the branch and bound method([1] [7] [8] [10] [16] and possibly others) which is particularly useful to the problem of (0, 1)-valued variables. One of the remarkable properties of integer programming is the erratic convergence speed of a method, depending upon the type of a particular problem. A method which can solve efficiently one problem may not be efficient for others. Probably each particular problem has a suitable method and furthermore the method can be modified so that the convergence is sped up.

An integer linear programming approach was first brought into the design of an optimum feed-forward network of threshold gates by S. H. Cameron in 1964[5]. His formulation, however, needs an exponentially increasing number of variables as the number R of gates in a network increases, while our approach uses the number of variables in the order of R^2. Cameron's work was followed by M. Breuer's[4] which considerably simplified the formulation. Our work which is essentially the same approach as Breuer's but discussed more comprehensively various aspects of the formulation was initiated independently (the first presentation was [20]) and needed

slightly fewer inequalities than Breuer's. The current paper is a revision of [20] and will be a basis of the further application of integer linear programming to logical design which will be published elsewhere[22].

In the following, let us first introduce our integer linear programming formulation based on a network of threshold gates. This is a general formulation because conventional switching gates such as AND, OR, NAND and NOR may be considered as special threshold gates with their weights and thresholds appropriately specified. As we will see in the succeeding paper[22], this general formulation will be a basis of logical design of more complicated networks such as a network where each gate may be chosen from a given set of different types of gates. (Accordingly a resultant network consists of a mixture of different types of gates.)

Logical design of an optimum network with only NOR gates will be discussed later as an example of special cases of the general formulation.

2. Definition

A threshold gate is defined as a logic gate in which each variable input x_ℓ ($\ell = 1, 2, \ldots, n$) is 1 or 0 and for which there exists a set of real numbers, $w_1, \ldots, w_n$, called weights and T called a threshold such that the output of the gate is;

$$\begin{aligned} &1 \text{ if } \sum_{\ell=1}^{n} w_\ell x_\ell^{(j)} \geq T \\ &0 \text{ if } \sum_{\ell=1}^{n} w_\ell x_\ell^{(j)} \leq T - 1, \quad (j = 1, 2, \ldots, m) , \end{aligned} \tag{2.1}*$$

where $x^{(j)} = (x_1^{(j)}, \ldots, x_n^{(j)})$ is the j-th input vector and $m(\leq 2^n)$ is the number of input vectors which specify the output values of the gate.

A feed-forward network is shown in Fig. 1 in which all threshold gates are arranged in a line and each threshold gate in the network can receive the inputs only from the external variables $x_1, x_2, \ldots, x_n$ or from outputs of the preceding gates on its left.

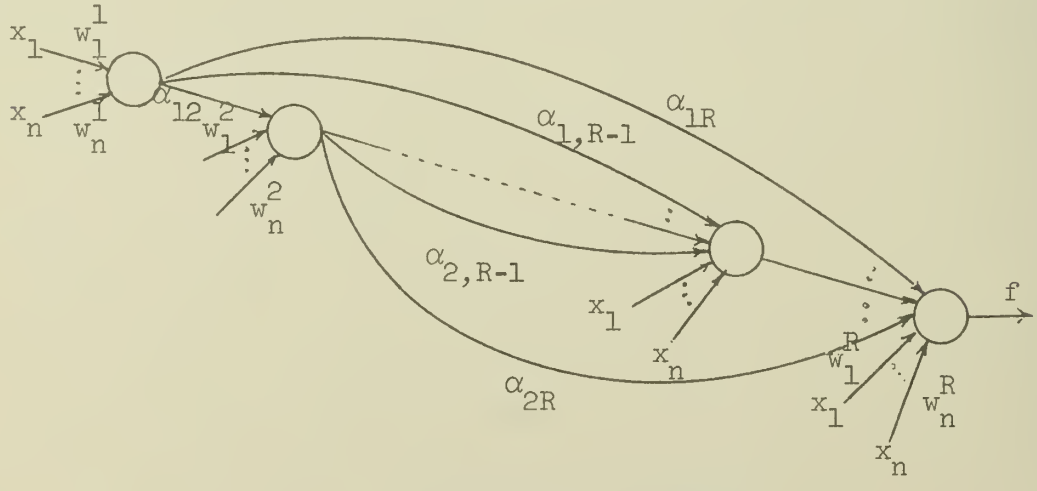

Fig. 1 A Feed-Forward Network

---

* Strictly speaking, these are called the normalized system of inequalities of a threshold gate and are derived by taking into account the margin of operation. For detail, see[21].

The k-th gate in the network has the weight vector.

$$\vec{w}^k = (w_1^k, w_2^k, \ldots, w_n^k)$$

for the external variables $x_1, x_2, \ldots, x_n$, and a threshold T_k. The weight of interconnection from the i-th gate to the k-th gate ($i < k$) is denoted by α_{ik}. The output value of the k-th gate for the j-th input vector $\vec{x}^{(j)}$ is denoted as $P_k^{(j)}$ and determined by

$$P_k^{(j)} = 1 \text{ if } \sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \alpha_{ik} P_i^{(j)} \geq T_k$$

$$P_k^{(j)} = 0 \text{ if } \sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \alpha_{ik} P_i^{(j)} \leq T_k - 1 \quad . \qquad (2.2)^*$$

Fig. 1 has R gates and the final gate is supposed to emit the output whose value for each input vector is specified in advance. In other words, if the network is to realize a switching function f, then

$$P_R^{(j)} = f(\vec{x}^{(j)}) \qquad (2.3)$$

must hold for each $\vec{x}^{(j)}$.

A feed-forward network is the most general case of a loop-free network. Henceforth, we will consider only feed-forward networks with respect to our integer linear programming approach.

---

* $\sum_{i=1}^{0}$ is defined to be o so that the inequality expression (2.2) may be applicable to the first gate.

3. Description of a feed-forward network in integer linear inequalities.

Here we completely characterize the operation of a feed-forward network by linear inequalities with some variables restricted to be integers. In Section 5, a synthesis procedure of an optimal network based on this formulation will be given.

Let us consider the k-th gate in a feed-forward network shown in Fig. 1. A set of new variables $\beta_{ik}^{(j)}$ is introduced in association with α_{ik} and $P_i^{(j)}$, such that

$$\beta_{ik}^{(j)} = \alpha_{ik} P_i^{(j)} \quad . \tag{3.1}$$

$\beta_{ik}^{(j)}$ may be considered as the input value from the i-th gate to the k-th gate for the j-th vector. The non-linear relation (3.1) will be rewritten in linear inequalities later. Then from (2.2), the output of the k-th gate is defined as follows:

$$\begin{aligned} P_k^{(j)} &= 1 \quad \text{if} \sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \beta_{ik}^{(j)} \geq T_k \\ P_k^{(i)} &= 0 \quad \text{if} \sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \beta_{ik}^{(j)} \leq T_k - 1 \quad . \end{aligned} \tag{3.2}$$

However, these conditional relations can be converted into a set of linear inequalities.

$$\sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \beta_{ik}^{(j)} - T_k \geq - U(1 - P_k^{(j)}), \tag{3.3A}$$

$$- \sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} - \sum_{i=1}^{k-1} \beta_{ik}^{(j)} + T_k - 1 \geq - UP_k^{(j)} \quad , \tag{3.3B}$$

where $P_k^{(j)}$ is 1 or 0.

Let us assume that U is a sufficiently large positive number.

Then if $P_k^{(j)} = 1$, then (3.3A) may be obviously rewritten as

$$\sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{k=1}^{k-1} \beta_{ik}^{(j)} - T_k \geq 0$$

and (3.3B) becomes non-restrictive. Otherwise, i.e., if $P_k^{(j)} = 0$, (3.3B) may be rewritten as

$$-\sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} - \sum_{i=1}^{k-1} \beta_{ik}^{(j)} + T_k - 1 \geq 0$$

and (3.3A) becomes now non-restrictive. Therefore (3.3A) and (3.3B) together are equivalently used as (3.2).

For the above purpose, U is large enough if it is not smaller than the conceivable maximum of the absolute value of the left hand side of each inequality. For (3.3A), for instance, U satisfies

$$\begin{aligned} U &\geq \sum_{\ell=1}^{m} | w_\ell^k | + \sum_{i=1}^{k-1} | \alpha_{ik} | + | T_k | \\ &= \sum_{\ell=1}^{} | w_\ell^k | + \sum_{i=1}^{k-1} | \beta_{ik}^{(j)} | + | T_k | \qquad (3.4) \\ &\geq | \sum_{\ell=1}^{m} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \beta_{ik}^{(j)} - T_k | \end{aligned}$$

is satisfactory. An upper bound of a weight and a threshold of d input threshold gate is known[18] [21] and shown below.

$$\begin{aligned} | w_i | &\leq 2\left(\frac{d+1}{4}\right)^{\frac{d+1}{2}} \\ | T | &\leq 2d\left(\frac{d+1}{4}\right)^{\frac{d+1}{2}} . \end{aligned} \qquad (3.5)$$

Considering the k-th gate as a threshold gate with (n + k - 1) inputs

and using (3.4), U which is not smaller than

$$4(n + k - 1)\left(\frac{n+k}{4}\right)^{\frac{n+k}{2}} \tag{3.6}$$

is sufficient for (3.3A).

Henceforth we will use the letter U exclusively for this purpose. We will use the same U for all inequalities, though U may assume a different value for a different inequality*. A lower bound of U for each inequality in this paper will be obtained in a similar way as shown above. The derivation of each U will be omitted.

We have introduced all necessary variables in the above. Note that $P_k^{(j)}$ is only variables which are restricted to be integers. All other variables are real numbers which are possibly negative. The integer linear programming formulation, however, requests all variables being non-negative (see (1.1)). This can be solved by splitting each variable as follows:

$$\begin{aligned} w_\ell^k &= w_\ell^{k+} - w_\ell^{k-} \\ \alpha_{ik} &= \alpha_{ik}^{+} - \alpha_{ik}^{-} \\ \beta_{ik}^{(j)} &= \beta_{ik}^{(j)+} - \beta_{ik}^{(j)-}, \end{aligned} \tag{3.7}$$

$$(k = 1, 2, \ldots, R)$$
$$(i = 1, 2, \ldots, k - 1)$$
$$(\ell = 1, 2, \ldots, n)$$
$$(j = 1, 2, \ldots, m),$$

---

* For some algorithms of integer linear programming such as Gomory's, the smallest possible U is preferable in order to avoid the overflow during the computation. The smallest values of U are computed for small numbers of inputs[23] [24].

with constraints

$$w_{\ell}^{k+} \geq 0 \ , \ w_{\ell}^{k-} \geq 0$$

$$\alpha_{ik}^{+} \geq 0, \ \alpha_{ik}^{-} \geq 0$$

$$\beta_{ik}^{(j)+} \geq 0, \ \beta_{ik}^{(j)-} \geq 0 \quad . \tag{3.8}$$

For notational symplicity, however, w_{ℓ}^{k} etc. will be usually used in place of $w_{\ell}^{k+} - w_{\ell}^{k-}$ etc., unless otherwise specified.

Now that output of threshold gate is described by (3.3), the non-linear equality (3.1) may be expressed in linear inequalities as follows:

$$\beta_{ik}^{(j)} \leq \alpha_{ik} + U(1 - P_i^{(j)}), \tag{3.9A}$$

$$\alpha_{ik} \leq \beta_{ik}^{(j)} + U(1 - P_i^{(j)}), \tag{3.9B}$$

$$\beta_{ik}^{(j)+} + \beta_{ik}^{(j)-} \leq UP_i^{(j)}, \tag{3.9C}$$

$$(i = 1, 2, \ldots, R - 1)$$

$$(k = i + 1, i + 2, \ldots, R)$$

$$(j = 1, 2, \ldots, m).$$

It can be easily seen that if $P_i^{(j)} = 1$, then all three inequalities of (3.9) lead to

$$\beta_{ik}^{(j)} = \alpha_{ik}$$

and if $P_i^{(j)} = 0$, then

$$\beta_{ik}^{(j)} = 0 \ ,$$

holds, because of $\beta_{ik}^{(j)+} \geq 0$, $B_{ik}^{(j)-} \geq 0$ and (3.9.C). This is exactly what (3.1) implies.

The inequalities (3.9.C) for different k, i.e., k = i + 1, i + 2, ..., R, can be packed into the following single inequality,

$$\sum_{k=i+1}^{R} (\beta_{ik}^{(j)+} + \beta_{ik}^{(j)-}) \leq UP_i^{(j)}, \qquad (3.10)$$

$$(i = 1, 2, \ldots, R - 1)$$

$$(j = 1, 2, \ldots, m),$$

reducing the number of inequalities. This is possible because when the output of the i-th gate $P_i^{(j)}$ is 0, $\beta_{ik}^{(j)} = 0$ for k = i + 1, i + 2, ..., R must follow.

(3.3) and (3.9) together characterize the entire feed-forward network except the last gate (i.e., the Rth gate). The output of the last gate is specified by (2.3). By specifying the value of $P_R^{(j)}$ in (3.3) according to (2.3), the inequalities for the R-th gate

$$\sum_{\ell=1}^{n} w_{\ell}^{R} x_{\ell}^{(j)} + \sum_{i=1}^{R-1} \beta_{iR}^{(j)} - T_R \geq 0 \quad \text{for} \quad f(\vec{x}^{(j)}) = 1$$

$$-\sum_{\ell=1}^{n} w_{\ell}^{R} x_{\ell}^{(j)} - \sum_{i=1}^{R-1} \beta_{iR}^{(j)} + T_R - 1 \geq 0 \quad \text{for} \quad f(\vec{x}^{(j)}) = 0, \qquad (3.11)$$

$$j = 1, 2, \ldots, m,$$

are obtained.

In conclusion, a feed-forward network as shown in Fig. 1 is described by the following inequalities with part of the variables constrained to be integers.

For the k-th gate, k = 1, 2, ..., R - 1,

$$\sum_{\ell=1}^{n} w_{\ell}^{k} x_{\ell}^{(j)} + \sum_{i=1}^{k-1} \beta_{ik}^{(j)} - T_k \geq - U(1-P_k^{(j)})$$

$$-\sum_{\ell=1}^{n} w_{\ell}^{k} x_{\ell}^{(j)} - \sum_{i=1}^{k-1} \beta_{ik}^{(j)} + T_k - 1 \geq - UP_k^{(j)} \qquad (3.12)$$

$$(j = 1, 2, \ldots, m)$$

$$\beta_{ik}^{(j)} \leq \alpha_{ik} + U(1 - P_i^{(j)})$$

$$\alpha_{ik} \leq \beta_{ik}^{(j)} + U(1 - P_i^{(j)}) , \qquad (3.13)^*$$

$$(i = 1, 2, \ldots, k - 1)$$

$$(j = 1, 2, \ldots, m) ,$$

$$\sum_{i=k+1}^{R} (\beta_{ki}^{(j)+} + \beta_{ki}^{(j)-}) \leq UP_k^{(j)}, \qquad (3.14)$$

$$(j = 1, 2, \ldots, m) .$$

For the R-th gate,

$$\sum_{\ell=1}^{n} w_\ell^R x_\ell^{(j)} + \sum_{i=1}^{R-1} \beta_{iR}^{(j)} - T_R \geq 0 \quad \text{for } f(\vec{x}^{(j)}) = 1$$

$$-\sum_{\ell=1}^{n} w_\ell^R x_\ell^{(j)} - \sum_{i=1}^{R-1} \beta_{iR}^{(j)} + T_R - 1 \geq 0 \quad \text{for } f(\vec{x}^{(j)}) = 0 , \qquad (3.15)$$

$$(j = 1, 2, \ldots, m) ,$$

$$\beta_{iR}^{(j)} \leq \alpha_{iR} + U(1 - P_i^{(j)})$$

$$\alpha_{iR} \leq \beta_{iR}^{(j)} + U(1 - P_i^{(j)}) , \qquad (3.16)$$

$$(i = 1, 2, \ldots, R - 1)$$

$$(j = 1, 2, \ldots, m).$$

Here $w_\ell^k, \beta_{ik}^{(j)}, \alpha_{ik}$, where $k = 1, 2, \ldots, R$, $i = 1, 2, \ldots, k-1$, $\ell = 1, 2, \ldots, n$, and $j = 1, 2, \ldots, m$, are real numbers which are actually represented as $w_\ell^k = w_\ell^{k+} - w_\ell^{k-}$ and so forth. The variables $P_i^{(j)}$ are integers whose values are 0 or 1**.

---

* The inequality for k=1 is not needed in (3.13)

** Inequalities $P_i^{(j)} \leq 1$ must be added when Gomory's algorithm is applied. In case of the implicit enumeration method, these are omitted.

The numbers of variables are

$$m(R - 1) \qquad \text{(integer variables)}$$

$$\text{plus} \qquad n\,R + (1 + m)\,\frac{R(R-1)}{2} \quad \text{(real variables)} \tag{3.17}$$

The number of inequalities is

$$m + 3m(R - 1) + mR(R - 1) \quad , \tag{3.18}$$

where

R: the number of gates in a network

m: the number of specified vectors

n: the number of external input variables.

4. Incorporation of network restrictions.

Some of the restrictions imposed on a network can be taken into account by simply adding inequalities. Ease of incorporation of restrictions is one of the characteristics of the integer linear programming approach.

First let us discuss the restriction on input tolerance[19]; i.e., the maximum permissible deviation of gate parameters such as weights, a threshold and signal magnitudes. The input tolerance δ for a threshold gate can be written as follows under appropriate assumptions[14]

$$\delta = \frac{\min_j \left| \sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \beta_{ik}^{(j)} \right|}{\sum_{\ell=1}^{n} |w_\ell^k|} \qquad (4.1)$$

Under other assumption a different equation may result, as discussed in[21]. One of the other cases will be given in Section 6 in conjunction with majority gate expression.

When a certain objective function is incorporated, it is always satisfied* by an optimum solution that

$$\min_j \left| \sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \beta_{ik}^{(j)} \right| = 1 \qquad (4.2)$$

and

$$|w_\ell^k| = w_\ell^{k+} + w_\ell^{k-} \quad . \qquad (4.3)$$

In this case, if the input tolerance is specified as L, the inequality

---

* For example, the minimization of objective function

$$\sum_{k=1}^{R} \left(\sum_{\ell=1}^{n} (w_\ell^{k+} + w_\ell^{k-}) + \sum_{i=1}^{k-1} (\alpha_{ik}^{+} + \alpha_{ik}^{-}) \right)$$

guarantees (4.2) and (4.3). See Section 5.

$$\sum_{\ell=1}^{n} (w_{\ell}^{k+} + w_{\ell}^{k-}) + \sum_{i=1}^{k-1} (\alpha_{ik}^{+} + \alpha_{ik}^{-}) \leq \frac{1}{L} \qquad (4.4)$$

for the k-th gate will assure that the input tolerance is satisfied.

The restriction on the maximum number of inputs for each gate, i.e., the maximum fan-ins, can also be considered. Let I be the specified maximum number of inputs and introduce the following variables whose values are 0 or 1,

$$\mu_{\ell}^{k} \leq 1 \qquad (4.5)^{*}$$

and

$$\nu_{ik} \leq 1 \quad ,$$

associated with w_{ℓ}^{k} and α_{ik} as follows:

$$w_{\ell}^{k+} + w_{\ell}^{k-} \leq U\mu_{\ell}^{k}$$

$$\alpha_{ik}^{+} + \alpha_{ik}^{-} \leq U\nu_{ik} \; . \qquad (4.6)$$

The fan-in restriction is written as

$$\sum_{\ell=1}^{n} \mu_{\ell}^{k} + \sum_{i=1}^{k-1} \nu_{ik} \leq I \qquad (4.7)$$

for the k-th gate (k = 1, 2, ..., R). $\mu_{\ell}^{k} = 0$, for example, means that w_{ℓ}^{k+} and w_{ℓ}^{k-} are both 0, (see (4.6)) resulting in no connection from the ℓ-th external input variable to the k-th gate. Accordingly (4.7) which allows at most I of μ_{ℓ}^{k} and ν_{ik} to assume 1 guarantees the

---

* These inequalities are not needed for the implicit enumeration method.

satisfaction of the given fan-in restriction.

The maximum number of outputs from each gate, the maximum fan-outs, can also be treated similarly.

Next, let us consider the restriction on the number of levels of a network. Let us assume a feed-forward network have R gates and let the maximum number of levels be J. Then there arise sets of interconnections which are not permitted because of the level restriction. For example,

$$\alpha_{12} \neq 0,\ \alpha_{23} \neq 0,\ \ldots,\ \alpha_{J,J+1} \neq 0,\ \alpha_{J+1,J+2} \neq 0 \qquad (4.8)$$

are not allowed because (J + 1) consecutive interconnections indicate existance of more than J levels. This set of interconnections can be prohibited by using the zero-one variables ν_{ik} defined by (4.5) and (4.6) as repeated in the following,

$$\begin{aligned} &\nu_{ik} \leq 1 \\ &\alpha^{+}_{ik} + \alpha^{-}_{ik} \leq U\nu_{ik}, \\ &\qquad\qquad (k = 2,\ 3,\ \ldots,\ R) \\ &\qquad\qquad (i = 1,\ 2,\ \ldots,\ k - 1). \end{aligned} \qquad (4.9)$$

The inequality

$$\nu_{12} + \nu_{23} + \cdots + \nu_{J+1,J+2} \leq J \qquad (4.10)$$

assures that at least one of α_{12}, α_{23}, ..., $\alpha_{J+1,J+2}$ is 0, thus destroying the consecutive interconnections from the first gate to the (J + 2)-th gate. Consequently a procedure to keep a network within J levels is*: (1) first exhaust all sets of interconnections which have

---

* Alternatively, we can employ a feed-next network instead of the feed-forward network as a basis, so that the level problem may automatically be avoided. See the reference [20].

more than J levels and (2) add all the corresponding inequalities derived in a similar way as (4.10).

Other types of restrictions on interconnections can usually be incorporated. For example, if an interconnection α_{ik} is prohibited for some reason, we can simply add

$$\alpha^{+}_{ik} = \alpha^{-}_{ik} = 0 . \tag{4.11}$$

The conditional restriction, i.e., the restriction that $\alpha_{i'k'} = 0$ if $\alpha_{ik} \neq 0$ holds, is realized by

$$\alpha^{+}_{i'k'} + \alpha^{-}_{i'k'} \leq U(1 - \nu_{ik}) \quad , \tag{4.12}$$

where ν_{ik} is the zero-one variable defined by (4.9). Namely, if $\alpha_{ik} \neq 0$, then $\nu_{ik} = 1$ by (4.9) and it forces $\alpha_{i'k'} = \alpha^{+}_{i'k'} - \alpha^{-}_{i'k'}$ to 0 by (4.12).

By making use of the above observations, more involved conditions such as the planarity of a network could be also incorporated. A procedure is first to list all conditions which lead to the non-planarity and then to set up corresponding inequalities to prevent them. However, if R and n increase, the number of inequalities will become excessively large, and this approach would be impractical.

One way to avoid this difficulty is: (1) solve the set of inequalities without adding the extra inequalities which guarantees the planarity. (2) If a solution obtained is planar, then it is an optimal solution, otherwise (3) add some of the inequalities from the extra inequalities explained above, so that the current solution may become infeasible. Then solve the new problem resulted. By repeating (3), we will eventually obtain an optimal planar solution.

When we solve a given integer linear programming problem by the implicit enumeration method, restrictions to be incorporated need not be in linear inequalities. Then algorithms to test the planarity of a network which have been known to date can be incorporated without being written in linear inequalities in the integer linear programming problem of Section 3. And a planar optimum solution will be obtained. This approach will be further extended and explored elsewhere.

5. Procedures to design optimum networks.

We have so far discussed only the constraints of the integer linear programming problem (1.1). Now the objective function $\vec{c}\,\vec{y}$ and the entire procedure for deriving an optimum network will be presented.

First consider the realization of a feed-forward network which has the minimum number of gates. As far as only the minimization of the number of gates is concerned, any expression can be used as an objective function for the following Procedure I. So for the moment let us assume that we can have an objective function $\vec{c}\,\vec{y}$ which is a linear function of variables. Concrete expressions for $\vec{c}\,\vec{y}$ will be discussed later in this section.

Procedure I

1) Set R = 1 (If we know a lower bound of the number of gates required for realizing a given function by some other means, set R to this number).

2) Formulate a set of inequalities for a R gate feed-forward network and possibly inequalities corresponding to the restrictions imposed. Solve this problem with an objective function $\vec{c}\ \vec{y}$ by using an appropriate algorithm of integer linear programming. If this has a solution, it is an optimal solution. Otherwise go to 3).

3) Increase R by 1 and return to 2).

The whole procedure will terminate in a finite number of steps if the problem is feasible for some R.

An integer linear programming problem in the above procedure is generally a mixed-integer integer linear programming problem. However, the problem can be converted into an all-integer integer linear programming problem by restricting all variables to integers. If a mixed-integer problem has a solution, so does the corresponding all-integer problem, though their solutions may be different. In some cases, solving the all integer problem may be easier than the mixed integer problem.

Now let us discuss concrete expressions for the objective function $\vec{c}\ \vec{y}$, when we solve Procedure I. Since Procedure I yields a network of a minimum number of gates for any expression for $\vec{c}\ \vec{y}$, we can minimize (or maximize) other parameter of a network at the same time, by choosing an appropriate expression for $\vec{c}\ \vec{y}$. In other words, Procedure I will yield a network which has the minimum number of gates and an

optimized parameter, as will be discussed in the following.

As discussed in Section 4, the mimimization of

$$\sum_{\ell=1}^{n} |w_{\ell}^{k}| + \sum_{i=1}^{k-1} |\alpha_{ik}| \qquad (5.1)$$

as the objective function $\vec{c}\,\vec{y}$ means the maximization of the input tolerance of the k-th gate defined under a certain definition [14]. As the input tolerance is a maximum permissible deviation of gate parameters such as weights and a threshold, it means the maximization of the reliability of operation of a gate. Therefore the minimization of the sum of (5.1) over all the gates, i.e.

$$\sum_{k=1}^{R} \left(\sum_{\ell=1}^{n} |w_{\ell}^{k}| + \sum_{i=1}^{k-1} |\alpha_{ik}| \right) \qquad (5.2)$$

means the maximization of the reliability of the whole network. (5.2) can be rewritten in a linear expression

$$\sum_{k=1}^{R} \left(\sum_{\ell=1}^{n} (w_{\ell}^{k+} + w_{\ell}^{k-}) + \sum_{i=1}^{k-1} (\alpha_{ik}^{+} + \alpha_{ik}^{-}) \right) \qquad (5.3)$$

since with the minimization of (5.3), the relation (4.3) is always guaranteed. Note that (4.2) also holds.

Under another assumption, the maximum input tolerance for the k-th gate can be attained by minimizing[21]

$$|T_k|$$

rather than minimizing (5.3). In this case,

$$\sum_{k=1}^{R}(T_k^+ + T_k^-) \tag{5.4}$$

will be the objective function to minimize. Also, with the objective function (5.4), $|T_k| = T_k^+ + T_k^-$ obviously hold.

Instead of the input tolerance, the minimization of the number of interconnections between gates can be taken into account by minimizing the objective function

$$\sum_{k=2}^{R}\sum_{i=1}^{k-1} \nu_{ik} \tag{5.5}$$

where ν_{ik} are the zero-one variables defined by (4.9). Note that with the minimization of (5.5), $\nu_{ik}=0$ means the non-existence of the interconnection α_{ik} while $\nu_{ik}=1$ means the existence.

The number of connections of the external input variables can also be considered by using the zero-one variables μ_ℓ^k defined by (4.5) and (4.6). The minimization of

$$\sum_{k=2}^{R}\sum_{i=1}^{k-1} \nu_{ik} + \sum_{k=1}^{R}\sum_{\ell=1}^{n} \mu_\ell^k \tag{5.6}$$

will yield a network with the minimum number of the whole interconnections including those from external inputs.

An alternative approach for deriving a feed-forward network with the minimum number of gates is outlined in the following.

Procedure II

First prepare a sufficiently large number of gates which will guarantee the realizability of a given function. Then introduce new zero-one variables λ_k such that

$$\sum_{h=k+1}^{R}(\alpha_{kh}^{+} + \alpha_{kh}^{-}) \leq U\,\lambda_k \quad (k=1,2,\ldots,R-1). \qquad (5.7)$$

As in the case of ν_{ik} and μ_{ℓ}^{k}, if there is at least one interconnection from the output of the k-th gate to any other gate, then λ_k assumes 1. If $\lambda_k = 0$, the output of the k-th gate is not connected to others. Thus the minimization of

$$\sum_{k=1}^{R-1} \lambda_k \qquad (5.8)$$

will result in the minimum number of gates which are actually used. Note that with the minimization of (5.8) the value of (5.8) shows the number of gates actually used except the last gate which always exists in order to produce the given function f.

Although this procedure needs only one integer linear program, the size of the problem is relatively larger than that of Procedure I. Judging from the fact that the convergence of most of integer linear programming algorithms become progressively slow with the increase of the size of problem, Procedure I may be preferred at present.

An upper bound of the number of gates in a feed-forward network which is necessary for realizing any function of n variables is known[17] [21] as follows:

$$\begin{cases} 2^{\frac{n-1}{2}} + 2^{\frac{n+1}{2}} + 1 & \text{for odd } n \geq 7 \\ 2^{\frac{n}{2}+1} + 1 & \text{for even } n \geq 6 , \end{cases} \tag{5.9}$$

which may be used as the number of gates in Procedure II.

An advantage of Procedure II is that the other types of objective functions such as discussed in conjunction with Procedure I can be used as the primary objective rather than the secondary. This can be done simply by replacing the objective function (5.8) by an objective function such as (5.3) or (5.6). This procedure may yield such networks as one with the minimum number of interconnections but with the number of gates which may not be minimum. For this prupose, therefore, the number of gates due to (5.9) may not be sufficient. However, to authors' knowledge, no theory is available about such number.

6. Feed-forward netwrok of majority gates

Instead of the threshold expression which has been used in the preceeding sections, the majority expression[21] is used often for a gate. n + 1 real numbers

$$w_o, w_1, \ldots\ldots, w_n \tag{6.1}$$

are associates with a single majority gate. A majority gate with n variable inputs operates as follows

$$\begin{aligned} &1 \text{ if } w_o + \sum_{\ell=1}^{n} w_\ell \xi_\ell^{(j)} \geq 1 \\ &-1 \text{ if } w_o + \sum_{\ell=1}^{n} w_\ell \xi_\ell^{(j)} \leq -1 , \end{aligned} \tag{6.2}$$

where $\vec{\xi}^{(j)} = (\xi_1^{(j)}, \ldots, \xi_n^{(j)})$ represents the j-th input vector with each component $\xi_\ell^{(j)}$ assuming -1 or 1 instead of 0 or 1. The k-th majority gate in a feed-forward network emits the output

$$\begin{aligned} &1 \text{ if } w_o^k + \sum_{\ell=1}^{n} w_\ell^k \xi_\ell^{(j)} + \sum_{i=1}^{k-1} \alpha_{ik} Q_i^{(j)} \geq 1 \\ &-1 \text{ if } w_o^k + \sum_{\ell=1}^{n} w_\ell^k \xi_\ell^{(j)} + \sum_{i=1}^{k-1} \alpha_{ik} Q_i^{(j)} \leq -1 , \end{aligned} \tag{6.3}$$

where α_{ik} is the weight of interconnection between the i-th gate and the j-th gate, and $Q_i^{(j)}$ is the output value of the i-th gate for the j-th input vector, which is also 1 or -1. Obviously (6.2) and (6.3) respectively correspond to (2.1) and (2.2) of the threshold expression. Compared

with the threshold expression in Section 2, the majority expression can describe more easily some properties of a switching gate. As far as the logical operation of a switching gate is concerned, both expressions posess no essential difference. But the input tolerance of a majority gate is respresented[17] [21] by

$$\frac{\min_j \left| w_o^k + \sum_{\ell=1}^{n} w_\ell^k \xi_\ell^{(j)} + \sum_{i=1}^{k-1} \alpha_{ik} Q_i^{(j)} \right|}{\sum_{\ell=0}^{n} |w_\ell^k|} \tag{6.4}$$

The set of inequalities which defines a feed-forward network with R majority gates is derived, analogously to that of Section 3.

Let

$$\gamma_{ik}^{(j)} = \alpha_{ik} Q_i^{(j)} \tag{6.5}$$

and

$$P_i^{(j)} = \frac{1+Q_i^{(j)}}{2} \quad , \tag{6.6}$$

where $P_i^{(j)}$ assumes 0 and 1 according to $Q_i^{(j)} = -1$ and 1 respectively. For the k-th majority gate, k = 1, 2, ..., R-1,

$$\begin{aligned} w_o^k + \sum_{\ell=1}^{n} w_\ell^k \xi_\ell^{(j)} + \sum_{i=1}^{k-1} \gamma_{ik}^{(j)} &\geq 1 - U\,(1-P_k^{(j)}) \\ -w_o^k - \sum_{\ell=1}^{n} w_\ell^k \xi_\ell^{(j)} - \sum_{i=1}^{k-1} \gamma_{ik}^{(j)} &\geq 1 - U\,P_k^{(j)}, \\ & (j = 1, 2, \ldots, m) \ , \end{aligned} \tag{6.7}$$

$$
\begin{aligned}
&\gamma_{ik}^{(j)} \leq \alpha_{ik} + U\,(1-P_i^{(j)}) \\
&\alpha_{ik} \leq \gamma_{ik}^{(j)} + U\,(1-P_i^{(j)}) \\
&\gamma_{ik}^{(j)} \leq -\alpha_{ik} + U\,P_i^{(j)} \\
&-\alpha_{ik} \leq \gamma_{ik}^{(j)} + U\,P_i^{(j)}, \qquad (i = 1, 2, \ldots, k-1) \\
&\qquad\qquad\qquad\qquad\qquad\quad (j = 1, 2, \ldots, m)\,.
\end{aligned}
\tag{6.8}
$$

For the last majority gate

$$
\begin{aligned}
&w_o^R + \sum_{\ell=1}^{n} w_\ell^R \xi_\ell^{(j)} + \sum_{i=1}^{R-1} \gamma_{iR}^{(j)} \geq 1 \text{ for } f(\vec{\xi}^{(j)}) = 1 \\
&-w_o^R - \sum_{\ell=1}^{n} w_\ell^R \xi_\ell^{(j)} - \sum_{i=1}^{R-1} \gamma_{iR}^{(j)} \geq 1 \text{ for } f(\vec{\xi}^{(j)}) = -1\,, \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (j = 1, 2, \ldots, m)\,,
\end{aligned}
\tag{6.9}
$$

$$
\begin{aligned}
&\gamma_{iR}^{(j)} \leq \alpha_{iR} + U\,(1-P_i^{(j)}) \\
&\alpha_{iR} \leq \gamma_{iR}^{(j)} + U\,(1-P_i^{(j)}) \\
&\gamma_{iR}^{(j)} \leq -\alpha_{iR} + UP_i^{(j)} \\
&-\alpha_{iR} \leq \gamma_{iR}^{(j)} + UP_i^{(j)}\,, \\
&\qquad\qquad\qquad\qquad (i = 1, 2, \ldots, R-1) \\
&\qquad\qquad\qquad\qquad (j = 1, 2, \ldots, m)\,.
\end{aligned}
\tag{6.10}
$$

Only $P_i^{(j)}$ are zero-one variables. Other variables w_ℓ^k, α_{ik}, $\gamma_{ik}^{(j)}$, $k = 1, 2, \ldots, R$, $\ell = 1, 2, \ldots, n$, $i = 1, 2, \ldots, k-1$, $j = 1, 2, \ldots, m$, are all real numbers which are actually written as

$$
\begin{aligned}
w_\ell^k &= w_\ell^{k+} - w_\ell^{k-} \\
\alpha_{ik} &= \alpha_{ik}^{+} - \alpha_{ik}^{-} \\
\gamma_{ik}^{(j)} &= \gamma_{ik}^{(j)+} - \gamma_{ik}^{(j)-}
\end{aligned}
\tag{6.11}
$$

in order to keep the non-negativeness. Variables $Q_k^{(j)}$ do not appear explicitly in the formulation. The set of inequalities (6.7), (6.8), (6.9) and (6.10) together completely specify a feed-forward network of majority gates. The rest of procedure for an optimal network is the same as that for threshold gates networks.

In this case, however, the restriction on the input tolerance discussed in Section 4 is

$$w_o^{k+} + w_o^{k-} + \sum_{\ell=1}^{n}(w_\ell^{k+} + w_\ell^{k-}) + \sum_{i=1}^{k-1}(\alpha_{ik}^{+} + \alpha_{ik}^{-}) \leq \frac{1}{L} \qquad (6.12)$$

rather than (4.4) since (6.4) is used as the input tolerance. A network with a maximum input tolerance can be obtained by replacing the objective function by

$$\sum_{k=1}^{R}(w_o^{k+} + w_o^{k-} + \sum_{\ell=1}^{n}(w_\ell^{k+} + w_\ell^{k-}) + \sum_{i=1}^{k-1}(\alpha_{ik}^{+} + \alpha_{ik}^{-})) \qquad (6.13)$$

(It is proved in [21] that the minimization of (6.13) leads to the maximization of the input tolerance of all gates.)

7. Multiple output feed-forward networks

One of the important but more difficult problems is the synthesis of a multiple output network in which more than one function is simultaneously realized. A feed-forward network of threshold gates generally with multiple outputs is displayed in Figure 2.

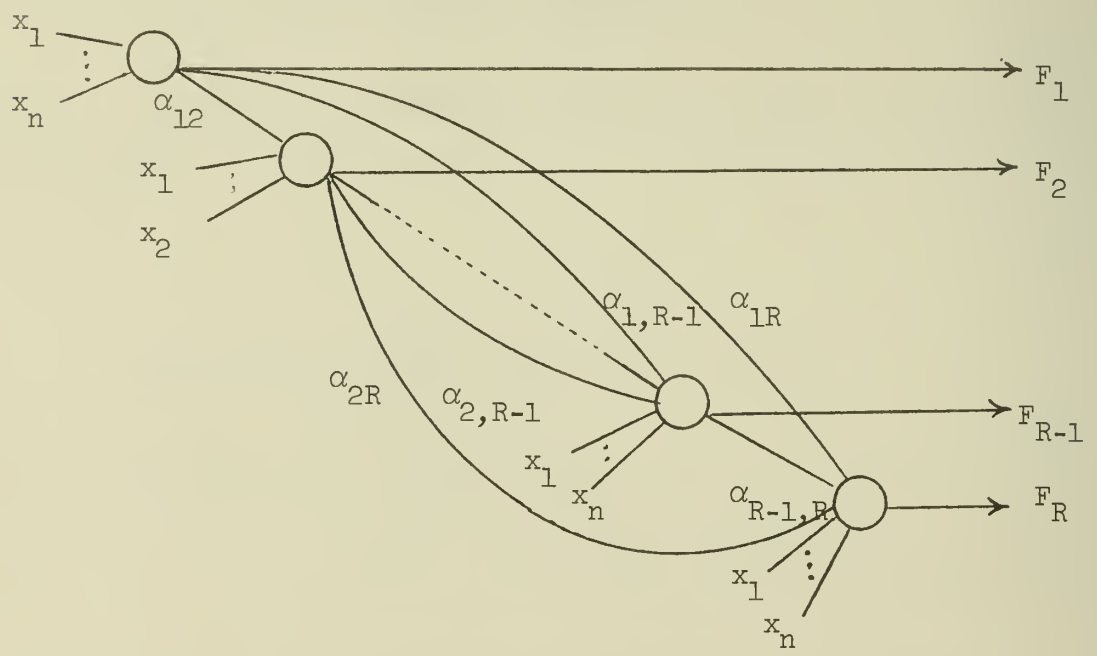

Fig. 2 A Feed-Forward Network With Multiple Outputs.

Each output from a gate in the network is denoted as F_k for the k-th gate, where

$$F_k(\vec{x}^{(j)}) = P_k^{(j)} \tag{7.1}$$

holds. $P_k^{(j)}$ is defined by (3.3) as in the case of a single output network.

When given S functions $f_1, f_2, \ldots, f_S$ are to be realized with the multiple output network of Fig. 2, each function must be realized by one of the F_k's. $(k = 1, 2, \ldots, R)$ If f_a is realized by F_b

$$f_a(\vec{x}^{(j)}) = F_b(\vec{x}^{(j)}) \qquad (j = 1, 2, \ldots, m_a) \tag{7.2}$$

must hold, where m_a is the number of specified vectors for f_a.

Let us introduce a set of new zero-one variables

$$\psi_{kt}, \qquad k = 1, 2, \ldots, R \qquad t = 1, 2, \ldots, S, \tag{7.3}$$

where $\psi_{kt} = 1$ if the output of the k-th gate realizes the function f_t, and $\psi_{kt} = 0$ otherwise. Then

$$\sum_{k=1}^{R} \psi_{kt} = 1, \quad t = 1, 2, \ldots, S \tag{7.4}$$

holds because each f_t is realized by exactly one output.

Now, if $\psi_{kt} = 1$, then

$$P_k^{(j)} = f_t(\vec{x}^{(j)}) \tag{7.5}$$

must follow from (7.1) and (7.2). This is realized by linear inequalities:

$$
\begin{aligned}
& f_t(\vec{x}^{(j)}) \leq P_k^{(j)} + U\,(1-\psi_{kt}) \\
& P_k^{(j)} \leq f_t(\vec{x}^{(j)}) + U(1-\psi_{kt}), \qquad (7.6) \\
& \qquad\qquad (t = 1, 2, \ldots, S) \\
& \qquad\qquad (k = 1, 2, \ldots, R) \\
& \qquad\qquad (j = 1, 2, \ldots, m_t)\,.
\end{aligned}
$$

Consequently the following set of inequalities characterizes a feed-forward network with R gates which realizes S given funstions. The rest of the network can be treated in the same way as the discussion in Section 3.

For the k-th threshold gate for all k = 1, 2, ..., R

$$
\begin{aligned}
& \sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \beta_{ik}^{(j)} - T_k \geq -U(1-P_k^{(j)}) \\
& -\sum_{\ell=1}^{n} w_\ell^k x_\ell^{(j)} - \sum_{i=1}^{k-1} \beta_{ik}^{(j)} + T_k - 1 \geq -UP_k^{(j)}, \qquad (7.7) \\
& \qquad\qquad (j = 1, 2, \ldots, m)^{*}\,,
\end{aligned}
$$

$$
\begin{aligned}
& \beta_{ik}^{(j)} \leq \alpha_{ik} + U(1-P_i^{(j)}) \\
& \alpha_{ik} \leq \beta_{ik}^{(j)} + U(1-P_i^{(j)})\,, \qquad (7.8) \\
& \qquad\qquad (i = 1, 2, \ldots, k-1) \\
& \qquad\qquad (j = 1, 2, \ldots, m),
\end{aligned}
$$

$$
\sum_{i=k+1}^{R} (\beta_{ki}^{(j)+} + \beta_{ki}^{(j)-}) \leq UP_k^{(j)}, \qquad (7.9)^{**}
$$

$$
(j = 1, 2, \ldots, m)\,,
$$

---

* j runs over all input vectors which are specified for at least one function of $f_1, f_2, \ldots, f_S$.

** $\sum_{R_1}^{R_2}$ with $R_1 > R_2$ is defined to be 0 to eliminate the inequality for k = R.

$$\sum_{k=1}^{R} \psi_{kt} = 1 \qquad (t = 1, 2, \ldots, S), \tag{7.10}*$$

$$\begin{aligned} f_t(\vec{x}^{(j)}) &\leq P_k^{(j)} + U(1-\psi_{kt}) \\ P_k^{(j)} &\leq f_t(\vec{x}^{(j)}) + U(1-\psi_{kt}), \end{aligned} \tag{7.11}$$

$$(k = 1, 2, \ldots, R)$$
$$(t = 1, 2, \ldots, S)$$
$$(j = 1, 2, \ldots, m_t) .$$

$P_k^{(j)}$ and ψ_{kt} are zero-one variables. Others are all real variables which should be split as, for example,

$$w_\ell^k = w_\ell^{k+} - w_\ell^{k-} . \tag{7.12}$$

Of course, $f_t(\vec{x}^{(j)})$ are given constants.

A procedure for a minimum gate synthesis is similarly performed as in Section 5 (of course a secondary objective can also be considered). Namely, starting from R = S (if all S functions are distinct), increase R by one until a feasible solution is found. The first solution gives an optimal solution.

Other considerations in Section 5 are also valid.

---

* This may be given by two inequalities.

8. Synthesis of an optimum network with NOR gates

As a special case of the previous discussion, a single output NOR gate network will be considered. This is important for the practical computer logical design. An NAND network may be similarly treated, though no discussion will be given. The use of more than one type of gate, such as a mixture of NOR and AND gates will be discussed in the succeeding paper [22]. Some of preliminary computational results will also be given in the next section.

The Boolean expression of an NOR gate for n variables $x_1, x_2, \ldots, x_n$ is

$$\overline{x_1 \vee x_2 \vee \ldots \vee x_n} = \bar{x}_1 \cdot \bar{x}_2 \ldots \bar{x}_n \quad . \tag{8.1}$$

In other words, if at least one of inputs is 1 then the output is 0, otherwise (i.e., all are 0) the output is 1. This is a threshold function with weights

$$\vec{w} = (-1, -1, \ldots, -1) \tag{8.2}$$

and threshold 0.

In our formulation of a network of NOR gates, all the variables assume only 1 or 0 which respecitvely represents the existence or non-existence of interconnections between gates and from external variable inputs. Our computational experience which will be outlined in the next section indicates that the implicit enumeration method is more suitable to our all-integer linear programming problem than Gomory's algorithms.

Let us define variables as follows:

v_ℓ^k: the interconnection to the k-th gate from the ℓ-th external variable input. $v_\ell^k = 1$ means the existence of the interconnection and $v_\ell^k = 0$ means the non-existence.

φ_{ik}: the interconnection from the i-th gate to the k-th gate. $\varphi_{ik} = 1$ means the existence of the interconnection and otherwise the non-existence.

$P_k^{(j)}$: the output value of the k-th gate for the j-th input vector.

$\rho_{ik}^{(j)}$: the input value to the k-th gate from the i-th gate for the j-th input vector.

Similarly to a general threshold gate, the k-th gate, k = 1, 2, ..., R-1, in a feed-forward network with R NOR gates can be described by

$$
\begin{aligned}
&- \sum_{\ell=1}^{n} v_\ell^k x_\ell^{(j)} - \sum_{i=1}^{k-1} \rho_{ik}^{(j)} \geq - U\,(1 - P_k^{(j)}) \\
&\sum_{\ell=1}^{n} v_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \rho_{ik}^{(j)} \geq 1 - U P_k^{(j)} , \\
&\qquad\qquad (j = 1, 2, \ldots, m) .
\end{aligned}
\tag{8.3}
$$

(8.3) implies that if at least one of $v_\ell^k x_\ell^{(j)}$ (the input value from the external variable input) and $\rho_{ik}^{(j)}$ is one, the output of the k-th gate is 0, and otherwise it is 1.

The input from other gates satisfies the relation

$$
\rho_{ik}^{(j)} = P_i^{(j)} \varphi_{ik} , \tag{8.4}
$$

which is equivalently written in the following set of inequalities

$$-P_i^{(j)} - \varphi_{ik} + \rho_{ik}^{(j)} \geq -1$$

$$P_i^{(j)} + \varphi_{ik} - 2\rho_{ik}^{(j)} \geq 0 , \qquad (8.5)$$

$$(k = 2, 3, \ldots, R)$$
$$(i = 1, 2, \ldots, k-1)$$
$$(j = 1, 2, \ldots, m).$$

(8.5) corresponds to (3.13) and (3.14). Since φ_{ik} and $\rho_{ik}^{(j)}$ are zero-one variables, while the α_{ik} and $\beta_{ik}^{(j)}$ in (3.13) and (3.14) are generally not, two inequalities are sufficient to specify the behavior of (8.4).

The last gate is treated separately since the output is specified by the given function f.

$$\begin{aligned} &- \sum_{\ell=1}^{n} v_\ell^k x_\ell^{(j)} - \sum_{i=1}^{k-1} \rho_{ik}^{(j)} \geq 0 \quad \text{for } f(\vec{x}^{(j)}) = 1 \\ &\sum_{\ell=1}^{n} v_\ell^k x_\ell^{(j)} + \sum_{i=1}^{k-1} \rho_{ik}^{(j)} \geq 1 \quad \text{for } f(\vec{x}^{(j)}) = 0 , \\ &\qquad j = 1, 2, \ldots, m. \end{aligned} \qquad (8.6)$$

All inequalities (8.3) (8.5) and (8.6) characterize the entire feed-forward network. The tested procedure of designing an optimal network is the same as Procedure I of Section 5 except that the number of interconnections

$$\sum_{k=1}^{R} \left(\sum_{\ell=1}^{n} v_\ell^k + \sum_{i=1}^{k-1} \varphi_{ik} \right) \qquad (8.7)$$

is used as the objective function since it is important in practice. Note that the expression for the number of interconnections needs more involved argument (see (5.6)) in the case of general threshold gate network.

When we solved this all-integer integer linear programming with the implicit enumeration method, we observed a very explicit general tendency that the smaller the solution space, the faster the termination of the computation. Therefore, efforts were made to reduce the size of the actual solution space to be calculated without losing the generality of the problem. In other words we added constraints so that solutions which are obviously not optimum or those which can be easily obtained from other solutions by permuting variables are suppressed. Important constraints among them will be briefly discussed in the following without proof.

(1) If the k-th gate and the last gate are connected, then it is proved that other outputs from the k-th gate have no contribution to the output of the last gate. (Fig. 3) In other words, if

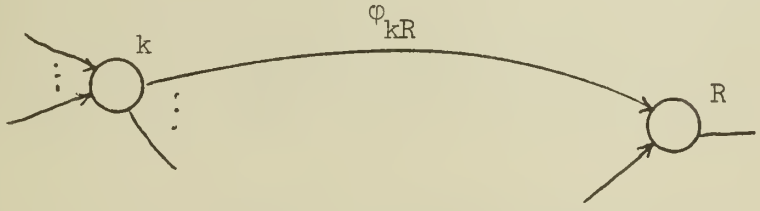

Fig. 3. Redundant Connections

$\varphi_{kR} = 1$, then we can automatically set

$$\varphi_{kk+1} = \varphi_{kk+2} = \cdots = \varphi_{kR-1} = 0 \ . \qquad (8.8)$$

(2) If the i-th gate is connected to the k-th gate (no other output connection from the i-th gate) and the k-th gate has only one input connection as shown in Fig. 4,

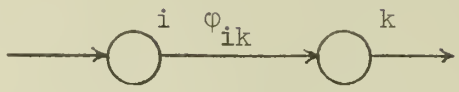

Fig. 4 Cascaded Connection

then the input value to the i-th gate and the output from the k-th gate have the same value. Thus both gates could be eliminated from the network.

(3) Suppose that the i-th gate is connected to the k-th gate ($i < k$) but no other output from the i-th gate exists. Then it is unnecessary that both gates to have inputs from the same external variables.

(4) Considering constraint (1), it is possible without loss of generality to impose an ordering on the connection to the last gate such that

$$\alpha_{R-1R} \geq \alpha_{R-2R} \geq \cdots \geq \alpha_{1R} \tag{8.9}$$

(5) Some of geometrical symmetries were also taken into consideration. For example, if the (R-2)-nd and (R-1)-st gates are connected

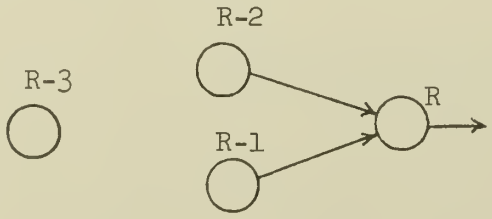

Fig. 5. Symmetry Connection

to the last gate and $\varphi_{R-2,R-1} = 0$ (this is in fact guaranteed by constraint (1)) then the (R-1)-st and (R-2)-nd gates are symmetric to the rest of preceeding gates. This justifies a new ordering such that

$$\varphi_{R-3,\ R-1} \geq \varphi_{R-3,\ R-2} \ . \tag{8.10}$$

This means that the (R-3)rd gate is connected to the (R-1)st gate before it is connected to the (R-2)nd gate. This type of symmetry was extensively studied with other gates also. However the details are omitted here.

(6) Other types of constraints were also incorporated such that each gate has at least one input connected and also at least one output when all R gates are assumed to be necessary.

All these constraints can be expressed in inequalities[3].

These inequalities in addition to the original set of inequalities which characterize a feed-forward network has significantly reduced the computation time. Some results will be reported in the next section.

9. Computational experience of designing optimum networks with NOR gates

Integer linear programming for designing an optimum feed-forward network of NOR gates in Section 8 was implemented on the IBM 360/75 and the ILLIAC II of the University of Illinois. All three variable Boolean functions are synthesized and compared with Hellerman's result by the exhaustive method[13].

Table 1 shows the size of problem for each R excluding additional inequalities discussed in the second half of Section 8.

Table 1 Sizes of Integer Linear Programming Problems

R	Matrix size		Coefficients		
	Constraints	Variables	Total No.	Non-Zero *	% Non-Zero
2	40	23	960	140	14.58
3	88	52	4664	332	7.12
4	152	90	13832	596	4.31
5	232	137	32016	932	2.91
6	328	193	63632	1340	2.11
7	440	258	113960	1820	1.60

* These numbers fluctutate slightly with different Boolean functions. These numbers are for the function which is identically 1.

The majority of published reports on computational efficiency of integer linear programming algorithms have the number of variables greater than the number of inequalities. Our problem is opposite. The number of inequalities is far greater than the number of variables. As a result when the implicit enumeration method is applied, it is difficult to find a feasible solution of this problem but once found, it is usually close to the optimal solution, whereas most of the problems published in the literature of operations research are easy to get the first feasible solution but take long computation time to improve it until an optimal solution is found.

The non-zero coefficients are very few as seen from Table 1. This was fully utilized in programming to speed up the computation.

The number of additional inequalities used for the derivation of all optimal networks for 3 variable Boolean functions is shown in Table 2. Some of them are very effective in reducing the computation time but some others are not too effective.

R	no. of additional ineq.
4	17
5	33
6	87
7	276

Table 2. Number of Additional Inequalities

Gomory's all-integer integer linear programming algorithm[12] was first used.* The detailed result will be found in [2]. This approach was not satisfactory at least for our problem, nevertheless we solved

---

* This formulation needs such a set of inequalities as

$$v_{\ell}^{k} \leq 1,\ \varphi_{ik} \leq 1,\ \rho_{ik}^{(j)} \leq 1,\ P_{k}^{(j)} \leq 1$$

which are not included in Table 1. For the sake of computational efficiency, a slightly modified formulation was actually used [2].

considerably larger integer programming problems than previously reported attempts. Formulations for R = 3 networks were solved within 10 seconds on the average using the computer ILLIAC II. However, computation time for R = 4 networks were very erratic. Some of them were solved within 20 minutes but others were not.

Discarding Gomory's method, the implicit enumeration was implemented on the IBM 360/75 with FORTRAN IV. Several modifications and improvements were also incorporated. As seen in Tables 1 and 2, the number of inequalities increases as R increases. But we made modifications such that computation time is least affected by the number of inequalities. The detail will be included in future papers[3] [15].

Optimal NOR networks for all the 3 variable Boolean functions were obtained within 110 minutes, by solving our integer program as a general integer program [15]. This may be favorably compared with Hellerman's exhaustive method in which he had solved the same problem consuming about 26 hours of IBM 7090 computer time. The effect of additional inequalities was remarkable. Some problems for R = 6 formulation was speeded up in the factor of 6.5 in computation time.

Considerable speed-up, however, was obtained by a more sophisticated modification of the implicit enumeration method by taking into account the physical structure of a network [3]. For example, computation time for a 3 variable function of 6 NOR gates was reduced to 3 seconds by this modification from 33 seconds by the above general method.

We are currently trying to improve the computation time. And the above figures should not be regarded as the maximum attainable ones.

Advantage of the implicit enumeration method over Gomory's algorithm is not only computational speed but also ease of non-optimal solutions. In other words, whenever we stop computation, the best solution obtained by then is likely close to the optimum one.

This result not only confirms the computational feasibility of the integer programming approach in logical design but also should encourage its adoption in other fields.

10. Conclusion

A feed-forward network of threshold gates was formulated as an integer linear programming problem. A specific problem, a NOR gate network, was tested by existing integer linear programming algorithms, and shown to be computationally feasible. In the light of these results it is concluded that the integer linear programming formulation provides us with a new tool for optimization problems in logical design which otherwise seem impossible to solve theoretically. The approach has many advantages in incorporating various restrictions on a network, choosing various design objectives and treating a multiple output network.

A generalization of the approach as well as the formulation of a design method of a sequential network will be included in the succeeding paper [22].

AC NOWLEDGEMENT

The authors would like to thank C. R. Baugh and T. K. Liu for their cooperation in performing the experiment in Section 9 and for their helpful comments.

REFERENCES

[1] E. Balas: An additive algorithm for solving linear programs with zero-one variables, Operations Research, Vol. 13, No. 4, pp. 517-544, July-August, 1965.

[2] C. R. Baugh, T. Ibaraki and S. Muroga: Computational experience in all-integer, binary variable, integer programming problems using Gomory's all-integer algorithm, Report No. 259, Department of Computer Science, University of Illinois, April 1968.

[3] C. R. Baugh, T. K. Liu, T. Ibaraki and S. Muroga: An optimum network design of NOR and NOR-AND gates by integer programming, to be published.

[4] M. A. Breuer: Implementation of threshold nets by integer linear programming, IEEE TEC, Vol. EC-14, No. 6, pp. 950-952, Dec. 1965.

[5] S. H. Cameron: The generation of minimal threshold nets by an integer program, IEEE TEC, Vol. EC-13, No. 3, pp. 299-302, June 1964.

[6] A. Cobham, R. Fridshal and J. H. North: An application of linear programming to the minimization of Boolean functions, Proceedings of the Second Annual Symposium of Switching Circuit Theory and Logical Design, pp. 3-9, 1961.

[7] N. J. Driebeek: An algorithm for the solution of mixed integer programming problems, Management Science, Vol. 12, No. 7, pp. 576-587, March 1966.

[8] A. M. Geoffrion: Integer programming by implicit enumeration and Balas' method, SIAM Review, Vol. 9, No. 2, pp. 178-190, April 1967.

[9] J. F. Gimpel: The minimization of TANT networks, IEEE TEC, Vol. EC-16, No. 1, pp. 18-38, Feb. 1967.

[10] F. Glover: A multiplephase-dual algorithm for the zero-one integer programming problem, Operations Research, Vol. 13, No. 6, pp. 879-919, Nov-Dec 1965.

[11] R. E. Gomory: An algorithm for the mixed integer problem, The Rand Corporation, P-1885, June 23, 1960.

[12] R. E. Gomory: An all-integer integer programming algorithm, Industrial Scheduling, edited by J. R. Muth and G. L. Thompson, Prentice-Hall, 1963, pp. 193-206.

[13] L. Hellerman: A catalog of three-variable OR-invert and AND-invert logical circuits, IEEE TEC, Vol. EC-12, No. 3, pp. 198-223, June 1963.

[14] T. Ibaraki and S. Muroga: Adaptive linear classifier by linear programming, to be published.

[15] T. Ibaraki, C. R. Baugh, T. K. Liu and S. Muroga: An implicit enumeration program for zero-one integer programming, to be published.

[16] A. H. Land and A. G. Doig: An automatic method of solving discrete programming problems, Econometrica, Vol. 28, No. 3, pp. 497-520, July, 1960.

[17] S. Muroga: Logical elements on majority decision principle and complexity of their circuits, International Conference on Information Processing, UNESCO, June 1959, Publsihed by Columbia University Press (Japanese version was delivered at Fukuoka, May 1958).

[18] S. Muroga, I. Toda and S. Takasu: Theory of majority decision elements, Journal of the Franklin Institute, Vol. 271, pp. 376-418, 1961.

[19] S. Muroga: Majority logic and problems of probabilistic behavior, Self-Organizing Systems, published by Spartan Books, 1962, pp. 243-281.

[20] S. Muroga: Logical design of an optimum network by integer linear programming, Lecture Notes of Threshold Logic, 1965-1966 School Year. Also File No. 700, Digital Computer Laboratory, University of Illinois, July 11, 1966.

[21] S. Muroga: Threshold logic, Lecture Notes for EE 497 and EE 498, Department of Computer Science, University of Illinois, 1965-1966 school year, to be published.

[22] S. Muroga and T. Ibaraki: Logical design of an optimum network by integer linear programming--Part II, to be published.

[23] S. Muroga, T. Tsuboi and C. R. Baugh: Enumeration of threshold functions of eight variables, Department of Computer Science, University of Illinois, Report No. 245, August 1967.

[24] R. O. Winder: Enumeration of seven-argument threshold functions, IEEE TEC, Vol. EC-14, No. 3, pp. 315-325, June 1965.